sevenstaxSOAP/XML/HTTP

User Manual

Revision No.:	2.1
State:	Released
Author:	sevenstax GmbH

Initial version:	24/10/07
Last change:	12/03/09
Last Review:	09/12/08
Publication:	Public
Filename:	SOAP_XML_HTTP_UserManual

Copyright (c) 2009 by sevenstax GmbH
This document is an intellectual property of sevenstax GmbH. Unauthorized copying and distribution is prohibited.

Table of Contents

1 Overview....5

2 SOAP module....6

2.1 Overview....6
2.2 Public functions....6
2.2.1 stxSOAPSrv_Init....6
2.2.2 stxSOAPSrv_Reset....7
2.2.3 stxSOAPSrv_Tick....7
2.2.4 stxSOAPSrv_SetNameServer....7
2.2.5 stxSOAPSrv_ RegNotifyHndlFct 7
2.2.6 stxSOAPSrv_RegHndlFct_GetAttrValLen....8
2.2.7 stxSOAPSrv_RegHndlFct_GetAttrValLenRepeat....8
2.2.8 stxSOAPSrv_RegHndlFct_GetAttrVal....8
2.2.9 stxSOAPSrv_RegHndlFct_GetInputValLen....9
2.2.10 stxSOAPSrv_RegHndlFct_GetInputValLenRepeat....9
2.2.11 stxSOAPSrv_RegHndlFct_GetInputVal....10
2.2.12 stxSOAPSrv_RegHndlFct_GetArrayCnt....10
2.2.13 stxSOAPSrv_RegHndlFct_GetArrayCntRepeat....10
2.2.14 stxSOAPSrv_RegHndlFct_PutAttrVal....11
2.2.15 stxSOAPSrv_RegHndlFct_PutOutputValPkt....11
2.2.16 stxSOAPSrv_SendRequest....11
2.3 Notify codes....13
2.3.1 NC_SOAP_READY....13
2.3.2 NC_SOAP_ERR_TCP_TIMEOUT_CONN....13
2.3.3 NC_SOAP_ERR_TCP_TIMEOUT_DATA....13
2.3.4 NC_SOAP_ERR_TCP_DISCONNECTED....13
2.3.5 NC_SOAP_ERR_TCP_WRITE....14
2.3.6 NC_SOAP_ERR_HTTP_GEN....14
2.3.7 NC_SOAP_ERR_HTTP_PRS....14
2.3.8 NC_SOAP_ERR_XML_GEN....14
2.3.9 NC_SOAP_ERR_XML_PRS....14
2.3.10 NC_SOAP_ERR_SOAP....15
2.3.11 NC_SOAP_TIMEOUT....15
2.4 SOAP instance module....16
2.4.1 XML tag list....16
2.4.2 XML attribute list....16
2.4.3 SOAP operation list....17

2.4.4 SOAP server namespace....20
2.4.5 sending XML prefix....20

3 XML module....21
3.1 Overview....21
3.2 Main part: Public functions....21
3.2.1 stxXML_RegNamespaces....21
3.3 Generator: Public functions....22
3.3.1 stxXMLGen_Init....22
3.3.2 stxXMLGen_SetEncoding....22
3.3.3 stxXMLGen_RegHndlFct_GetAttrValLen....22
3.3.4 stxXMLGen_RegHndlFct_GetAttrValLenRepeat....23
3.3.5 stxXMLGen_RegHndlFct_GetAttrVal....23
3.3.6 stxXMLGen_RegHndlFct_GetContentLen....23
3.3.7 stxXMLGen_RegHndlFct_GetContentLenRepeat....24
3.3.8 stxXMLGen_RegHndlFct_GetContent....24
3.3.9 stxXMLGen_RegHndlFct_GetArrayCnt....24
3.3.10 stxXMLGen_RegHndlFct_GetArrayCntRepeat....25
3.3.11 stxXMLGen_GetHeaderDataLen....25
3.3.12 stxXMLGen_GetHeaderData....25
3.3.13 stxXMLGen_GetCodeLen....26
3.3.14 stxXMLGen_GetCode....26
3.4 Parser: Public functions....27
3.4.1 stxXMLPrs_Init....27
3.4.2 stxXMLPrs_RegHndlFct_PutTagOpen....27
3.4.3 stxXMLPrs_RegHndlFct_PutAttrVal....27
3.4.4 stxXMLPrs_RegHndlFct_PutTagClose....28
3.4.5 stxXMLPrs_RegHndlFct_PutContentPkt....28
3.4.6 stxXMLPrs_RegHndlFct_ErrorHandler....28
3.4.7 stxXMLPrs_ParseInit....29
3.4.8 stxXMLPrs_Parse....29

4 HTTP module....30
4.1 Overview....30
4.2 Generator: Public functions....30
4.2.1 stxHTTPGen_Init....30
4.2.2 stxHTTPGen_SetMethod....30
4.2.3 stxHTTPGen_SetResource....31
4.2.4 stxHTTPGen_SetContentLength....31
4.2.5 stxHTTPGen_SetContentType....31
4.2.6 stxHTTPGen_SetHost....32
4.2.7 stxHTTPGen_SetUsernameAndPassword....32

4.2.8 stxHTTPGen_GetSendHeaderData....32
4.3 Parser: Public functions....33
4.3.1 stxHTTPPrs_Init....33
4.3.2 stxHTTPPrs_SetHndlFct_SetHeaderInfo....33
4.3.3 stxHTTPPrs_ParseInit....33
4.3.4 stxHTTPPrs_Parse....34
4.3.5 stxHTTPPrs_GetLastAnswerCode....34

1 Overview

SOAP is a protocol for exchanging structured XML information in Web Services Systems. Web Services are defined as "a software system designed to support interoperable machine-to-machine interaction over a network" (by W3C). The main properties are:

- Continuous with international standards
  - XML, SOAP, HTTP are international standards of the W3C
  - compatible to .net, Perl, PHP, C/C++, Phython, etc.
- Complete device integration into IT infrastructure
  - for example over SOA (service oriented architecture)
- data exchange with established applications, e.g.:
  - Microsoft Dynamics, .net tools, Software AG Tamino
  - IBM WebSphere, DB2 Viper
  - Apache SOAP, Apache Axis
  - SAP NetWeaver platform, Sun Java J2EE, ONE platform

The SOAP module can be used to send requests to and receive responses from a SOAP server. So, the sevenstax module works as a SOAP client. The requests and responses are sent over HTTP. To do this, SOAP uses separate HTTP generator and HTTP parser modules. To generate and parse the XML content, separate XML generator and XML parser modules are used. Before interchanging data with the server , a TCP connection as client must be established. To do this, the functionality of the Data Connect Service is used.

2 SOAP module

2.1 Overview

The SOAP module can be used to send requests to and receive responses from a SOAP server. So, the module works as a SOAP client. It accomplishes all steps from TCP connection establishment over generating and sending the request to receiving and parsing the response.

The SOAP module is composed of two parts: a core part and a user specific part.

The following files are included:

- soapsrv.c (core)
- soapdefs.h (internal definitions)
- soapsrv.h (public functions, public definitions)
- soapinst.c (user specific data: XML tag list, XML attribute list, XML templates, ...)
- soapinst.h (user specific definitions: XML content indices, ...)

The following public functions and notify codes can be used from any SOAP controlling module. In the sevenstax library SOAP is controlled by the Web Service Communication module (WSC).

2.2 Public functions

2.2.1 stxSOAPSrv_Init

void stxSOAPSrv_Init (void)

Description

Initializes the module.

Calls init functions of the HTTP and XML generator and parser modules and registers callback functions and XML namespaces.

The function must be called once before using any other SOAP functions.

Parameter

-

Return Value

-

2.2.2 stxSOAPSrv_Reset

void stxSOAPSrv_Reset (void)

Description

Resets the module so that a new SOAP request can be executed. Mainly internal variables are reset.

Parameter

-

Return Value

-

2.2.3 stxSOAPSrv_Tick

void stxSOAPSrv_Tick (void)

Description

Keeps SOAP alive. Handles timeout monitoring and TCP events

It must be called as often as possible.

Parameter

-

Return Value

-

2.2.4 stxSOAPSrv_SetNameServer

void stxSOAPSrv_SetSOAPServer
(STRING_stx szHost, UINT16_stx uPort, STRING_stx szResource)

Description

Defines the parameters of the SOAP server to be communicated with. Must be called to use SOAP.

The module saves the delivered data in internal variables, so that the delivered strings can be destroyed after the call.

Parameter

szHost:	Name of the SOAP server
uPort:	TCP port the SOAP server is listening on (often 80 for HTTP)
szResource:	SOAP resource to be connected with (argument of HTTP-POST)

Return Value

-

2.2.5 stxSOAPSrv_ RegNotifyHndlFct

void stxSOAPSrv_RegNotifyHndlFct
(PROTOCOL_NOTIFY_HANDLER fctAppNotifyPara)

Description

Registers a notify handler function to consume all SOAP specific notify codes (s.b.).

It is advisable to call this function before using SOAP.

Parameter

fctAppNotifyPara: Notify handler function
(PROTOCOL_NOTIFY_HANDLER is defined in “protdefs.h”)

Return Value

-

2.2.6 stxSOAPSrv_RegHndlFct_GetAttrValLen

void stxSOAPSrv_RegHndlFct_GetAttrValLen
(SOAP_GET_ATTRVAL_LEN_HNDL_FCT fHndlFct)

Description

Registers a handler function for requesting an actual attribute value length. The handler function is called during the sending process.

It is advisable to call this function before using SOAP.

Parameter

fHndlFct: Callback function
(SOAP_GET_ATTRVAL_LEN_HNDL_FCT is defined in “soapsrv.h”)

Return Value

-

2.2.7 stxSOAPSrv_RegHndlFct_GetAttrValLenRepeat

void stxSOAPSrv_RegHndlFct_GetAttrValLenRepeat
(SOAP_GET_ATTRVAL_LEN_HNDL_FCT fHndlFct)

Description

Registers a handler function for repeatedly requesting an actual attribute value length. The handler function is called during the sending process and MUST return the same value as in the former call of the function registered in the function stxSOAPSrv_RegHndlFct_GetAttrValLen().

It is necessary to freeze a value length to be sent because SOAP must calculate the exact content length in the request.

It is advisable to call this function before using SOAP.

Parameter

fHndlFct: Callback function
(SOAP_GET_ATTRVAL_LEN_HNDL_FCT is defined in “soapsrv.h”)

Return Value

-

2.2.8 stxSOAPSrv_RegHndlFct_GetAttrVal

void stxSOAPSrv_RegHndlFct_GetAttrVal

(SOAP_GET_ATTRVAL_HNDL_FCT fHndlFct)

Description

Registers a handler function for requesting an actual attribute value. The handler function is called during the sending process.

It is advisable to call this function before using SOAP.

Parameter

fHndlFct: Callback function (SOAP_GET_ATTRVAL_HNDL_FCT is defined in “soapsrv.h”)

Return Value

-

2.2.9 stxSOAPSrv_RegHndlFct_GetInputValLen

void stxSOAPSrv_RegHndlFct_GetInputValLen (SOAP_GET_INPUTVAL_LEN_HNDL_FCT fHndlFct)

Description

Registers a handler function for requesting an actual input value (content) length. The handler function is called during the sending process.

It is advisable to call this function before using SOAP.

Parameter

fHndlFct: Callback function (SOAP_GET_INPUTVAL_LEN_HNDL_FCT is defined in “soapsrv.h”)

Return Value

-

2.2.10 stxSOAPSrv_RegHndlFct_GetInputValLenRepeat

void stxSOAPSrv_RegHndlFct_GetInputValLenRepeat (SOAP_GET_INPUTVAL_LEN_HNDL_FCT fHndlFct)

Description

Registers a handler function for repeatedly requesting an actual input value (content) length. The handler function is called during the sending process and MUST return the same value as in the former call of the function registered in the function stxSOAPSrv_RegHndlFct_GetInputValLen().

It is necessary to freeze a value length to be sent because SOAP must calculate the exact content length in the request.

It is advisable to call this function before using SOAP.

Parameter

fHndlFct: Callback function (SOAP_GET_INPUTVAL_LEN_HNDL_FCT is defined in “soapsrv.h”)

Return Value

-

2.2.11 stxSOAPSrv_RegHndlFct_GetInputVal

void stxSOAPSrv_RegHndlFct_GetInputVal
(SOAP_GET_INPUTVAL_HNDL_FCT fHndlFct)

Description

Registers a handler function for requesting an actual input value (content). The handler function is called during the sending process.

It is advisable to call this function before using SOAP.

Parameter

fHndlFct: Callback function (SOAP_GET_INPUTVAL_LEN_HNDL_FCT is defined in “soapsrv.h”)

Return Value

-

2.2.12 stxSOAPSrv_RegHndlFct_GetArrayCnt

void stxSOAPSrv_RegHndlFct_GetArrayCnt
(SOAP_GET_ARRAY_CNT_HNDL_FCT fHndlFct)

Description

Registers a handler function for requesting an actual count of entries in an XML array. The handler function is called during the sending process. E.g. it can be used for sending log data.

It is advisable to call this function before using SOAP.

Parameter

fHndlFct: Callback function (SOAP_GET_ARRAY_CNT_HNDL_FCT is defined in “soapsrv.h”)

Return Value

-

2.2.13 stxSOAPSrv_RegHndlFct_GetArrayCntRepeat

void stxSOAPSrv_RegHndlFct_GetArrayCntRepeat
(SOAP_GET_ARRAY_CNT_HNDL_FCT fHndlFct)

Description

Registers a handler function for repeatedly requesting an actual count of entries in an XML array. The handler function is called during the sending process and MUST return the same value as in the former call of the function registered in the function stxSOAPSrv_RegHndlFct_GetArrayCnt(). E.g. it can be used for sending log data.

It is advisable to call this function before using SOAP.

Parameter

fHndlFct: Callback function (SOAP_GET_ARRAY_CNT_HNDL_FCT is defined in “soapsrv.h”)

Return Value

-

2.2.14 stxSOAPSrv_RegHndlFct_PutAttrVal

void stxSOAPSrv_RegHndlFct_PutAttrVal
(SOAP_PUT_ATTRVAL_PKT_HNDL_FCT fHndlFct)

Description

Registers a handler function for delivering a received attribute value. The handler function is called during the receiving process.

It is advisable to call this function before using SOAP.

Parameter

fHndlFct: Callback function
(SOAP_PUT_ATTRVAL_PKT_HNDL_FCT is defined in “soapsrv.h”)

Return Value

-

2.2.15 stxSOAPSrv_RegHndlFct_PutOutputValPkt

void stxSOAPSrv_RegHndlFct_PutOutputValPkt
(SOAP_PUT_OUTPUTVAL_PKT_HNDL_FCT fHndlFct);

Description

Registers a handler function for delivering a received output value (content). The handler function is called during the receiving process.

It is advisable to call this function before using SOAP.

Parameter

fHndlFct: Callback function
(SOAP_PUT_OUTPUTVAL_PKT_HNDL_FCT is def. in “soapsrv.h”)

Return Value

-

2.2.16 stxSOAPSrv_SendRequest

SOAP_ERR stxSOAPSrv_SendRequest
(CONN_HNDL_TYPE hConnLink, UINT8_stx uSOAPOperationIndexPara)

Description

Starts sending a SOAP request.

Parameter

hConnLink: (only in multi device version)
Handle of link layer connection (Ethernet / PPP / ...)

uSOAPOperationIndexPara: Index of SOAP operation which request is to be started.
A SOAP operation contains a request and response template. The list of all operations must be defined in “soapinst.c”.

Return Value

One of the following values:

SOAP_ERR_OK	Send order accepted
SOAP_ERR_BAD_INIT	Send order not accepted - user specific SOAP definitions are not correct (“soapinst.h”)
SOAP_ERR_BUSY	Send order not accepted - SOAP is busy

2.3 Notify codes

2.3.1 NC_SOAP_READY

Description

SOAP has finished successfully a request/response operation.

Parameter

-

Return Value

-

2.3.2 NC_SOAP_ERR_TCP_TIMEOUT_CONN

Description

SOAP has finished unsuccessfully.
A TCP connection timeout is occurred.

Parameter

-

Return Value

-

2.3.3 NC_SOAP_ERR_TCP_TIMEOUT_DATA

Description

SOAP has finished unsuccessfully.
During receiving TCP data the time difference between two packets was at least one minute.

Parameter

-

Return Value

-

2.3.4 NC_SOAP_ERR_TCP_DISCONNECTED

Description

SOAP has finished unsuccessfully.
An unexpected TCP disconnect is occurred.

Parameter

-

Return Value

-

2.3.5 NC_SOAP_ERR_TCP_WRITE

Description

SOAP has finished unsuccessfully.
Sending TCP data hat failed.

Parameter

-

Return Value

-

2.3.6 NC_SOAP_ERR_HTTP_GEN

Description

SOAP has finished unsuccessfully:
An error during generating HTTP code is occurred.

Parameter

-

Return Value

-

2.3.7 NC_SOAP_ERR_HTTP_PRS

Description

SOAP has finished unsuccessfully.
An error during parsing HTTP code is occurred.

Parameter

-

Return Value

-

2.3.8 NC_SOAP_ERR_XML_GEN

Description

SOAP has finished unsuccessfully.
An error during generating XML code is occurred.

Parameter

-

Return Value

-

2.3.9 NC_SOAP_ERR_XML_PRS

Description

SOAP has finished unsuccessfully.

An error during parsing XML code is occurred.

Parameter

-

Return Value

-

2.3.10 NC_SOAP_ERR_SOAP

Description

SOAP has finished unsuccessfully.
An XML tag that signalises a SOAP error was received.

Parameter

-

Return Value

-

2.3.11 NC_SOAP_TIMEOUT

Description

SOAP has finished unsuccessfully.
A SOAP request/response sequence has last more than 10 minutes.

Parameter

-

Return Value

-

2.4 SOAP instance module

In the SOAP instance module the user must define

- an XML tag list of his namespace
- an XML attribute list of his namespace
- a SOAP operation list
- a SOAP server namespace
- an XML prefix for sending tags of his namespace

2.4.1 XML tag list

The XML tag list of the user specific namespace must be defined in the array SOAPInst_TagList[]. Every entry consists of a structure that contains a string that defines the name of the tag. The structure of the last entry must contain a null pointer.

Example:

```
/* tag list names */
const XML_TAGLIST_ENTRY SOAPInst_TagList[] =
{
    {"data"},                   /*  0 */
    {"parameter"},              /*  1 */
    {"active"},                 /*  2 */
    ...
    {NULL_stx}
};
```

It is recommended that there is also a list of tag indices as macros to build the templates (s.b.).

2.4.2 XML attribute list

The XML attribute list of the user specific namespace must be defined in the array SOAPInst_AttrList[]. Every entry consists of a structure that contains a string that defines the name of the attribute. The structure of the last entry must contain a null pointer.

Example:

```
/* attribute list names */
const XML_ATTRLIST_ENTRY SOAPInst_AttrList[] =
{
    {"id"},                     /*  0 */
    {NULL_stx}
};
```

It is recommended that there is also a list of attribute indices as macros to build the templates (s.b.).

2.4.3 SOAP operation list

The SOAP operation list contains SOAP operations.

Every SOAP operation (type SOAP_OPERATION_TYPE) contains

- a SOAP action (used in SOAPAction field)
- a request XML template
- a response XML template

Every template is an array of XML entries (type XML_ENTRY). An XML entry contains an object type id, a namespace id and an object id.

The object type id is one of the following:

- `XML_MARK_TAG_OPEN` `/* tag open */`
- `XML_MARK_NS` `/* namespace declaration */`
- `XML_MARK_ATTR_NAME` `/* attribute name */`
- `XML_MARK_ATTR_VAL` `/* attribute value */`
- `XML_MARK_CONTENT` `/* content */`
- `XML_MARK_TAG_CLOSE` `/* tag close */`
- `XML_MARK_TAG_ARRAY_COUNT` `/* count of tag repetitions (array) */`
- `XML_MARK_END` `/* end of xml document */`

The namespace id must always be 0.

The object id depends of the object type id.

- `XML_MARK_TAG_OPEN /* tag open */`
  tag index (defined in “soapinst.c”)
- `XML_MARK_NS /* namespace declaration */`
  irrelevant
- `XML_MARK_ATTR_NAME /* attribute name */`
  attribute name index (defined in “soapinst.h”
- `XML_MARK_ATTR_VAL /* attribute value */`
  attribute value index (defined in “soapinst.h”)
- `XML_MARK_CONTENT /* content */`
  content index (defined in “soapinst.h”)
- `XML_MARK_TAG_CLOSE /* tag close */`
  irrelevant
- `XML_MARK_TAG_ARRAY_COUNT /* count of tag repetitions (array) */`
  array index (defined in “soapinst.h”)
- `XML_MARK_END /* end of xml document */`
  irrelevant

Example:

```
/*** list of all operations ***/
const SOAP_OPERATION_TYPE FPTR_stx uSOAPInst_Operations[] =
{
    &sSOAPInst_GETORDERS,
    &sSOAPInst_SENDDATA
};
```

with

```
/* operation struct */
const SOAP_OPERATION_TYPE sSOAPInst_GETORDERS =
{
    SOAPINST_ACTION,
    (XML_ENTRY FPTR_stx)rgSOAPInst_Request_GETORDERS,
    (XML_ENTRY FPTR_stx)rgSOAPInst_Response_GETORDERS
};
```

and

```
/** action name **/
#define SOAPINST_ACTION    MAKESTRING("urn:wechselrichter")
```

and

```
/* request xml template */
const XML_ENTRY rgSOAPInst_Request_GETORDERS[] =
{
    {XML_MARK_TAG_OPEN,          0,  TAG_GET_ORDERS},
    {XML_MARK_NS,                0,  _},

        {XML_MARK_TAG_OPEN,          0,  TAG_MID},
            {XML_MARK_CONTENT,           0,  CONTENT_MID},
        {XML_MARK_TAG_CLOSE,         0,  _},

        {XML_MARK_TAG_OPEN,          0,  TAG_CUSTOMER_CODE},
            {XML_MARK_CONTENT,           0,  CONTENT_C_CODE},
        {XML_MARK_TAG_CLOSE,         0,  _},

        {XML_MARK_TAG_OPEN,          0,  TAG_SENDALL},
            {XML_MARK_CONTENT,           0,  CONTENT_SENDALL},
        {XML_MARK_TAG_CLOSE,         0,  _},

        {XML_MARK_TAG_OPEN,          0,  TAG_ISP_USED},
            {XML_MARK_CONTENT,           0,  CONTENT_DATA_ISP_USED},
        {XML_MARK_TAG_CLOSE,         0,  _},

    {XML_MARK_TAG_CLOSE,         0,  _},

    {XML_MARK_END,               _,  _}
};
```

2.4.4 SOAP server namespace

Must be defined in the macro SOAPINST_XMLNS. It is used to generate the XML namespace declaration.

Example:

```
/* name of soap web service namespace */
#define SOAPINST_XMLNS          "urn:wechselrichter"
```

2.4.5 sending XML prefix

Must be defined in the macro SOAPINST_PREFIX. It is used to generate all XML tags and attributes.

Example:

```
/* sending prefix of sending tags & attributes of soap web service
namespace */
#define SOAPINST_PREFIX         ""
```

3 XML module

3.1 Overview

The XML module can be used to generate and parse XML documents.

XML code can be generated in segments of arbitrary length.

The parse process works “on the fly”. That means that the XML document can be parsed immediately during the receive process.

The XML module is composed of three parts: a main part, a generator and a parser.

The following files are included:

- xml.c (main part: structures and functions)
- xml.h (main part: function declarations, definitions)
- xmlgen.c (generator: functions)
- xmlgen.h (generator: function declarations, definitions)
- xmlprs.c (parser: functions)
- xmlprs.h (parser: function declarations, definitions)

The following public functions can be used from any XML controlling module. In the sevenstax library XML is controlled for instance by SOAP or REST.

3.2 Main part: Public functions

3.2.1 stxXML_RegNamespaces

void stxXML_RegNamespaces
(XML_NAMESPACE FPTR_stx FPTR_stx pNamespaces, UINT8_stx uCntNS)

Description

Registers a set of namespaces.

It must be called before using XML.

Parameter

pNamespaces: pointer to an array of namespaces
(XML_NAMESPACE is defined in “xml.h”)
The last two entries in every namespace (szNSPrefixReceived[], uValidityIndex[]) must have the values {NULL_stx} and {UINT8_stx_MAX}.

uCntNS: count of namespaces

Return Value

-

3.3 Generator: Public functions

3.3.1 stxXMLGen_Init

void stxXMLGen_Init(void)

Description

Initializes the module.

The function must be called once before using any other XML generator functions.

Parameter

-

Return Value

-

3.3.2 stxXMLGen_SetEncoding

void stxXMLGen_SetEncoding (UINT8_stx uEncoding)

Description

Sets the used XML encoding

Parameter

uEncoding: encoding to set, one of the following values:
XML_CHARSET_UTF8
XML_CHARSET_ISO8859_1

Return Value

-

3.3.3 stxXMLGen_RegHndlFct_GetAttrValLen

void stxXMLGen_RegHndlFct_GetAttrValLen
(XMLGEN_GET_ATTRVAL_LEN_HNDL_FCT fHndlFct)

Description

Function to register a handler function for delivering an attribute value length.

It is advisable to call this function before using XML generator.

Parameter

fHndlFct: handler function
(XMLGEN_GET_ATTRVAL_LEN_HNDL_FCT is defined in "xmlgen.h")

Return Value

-

3.3.4 stxXMLGen_RegHndlFct_GetAttrValLenRepeat

void stxXMLGen_RegHndlFct_GetAttrValLenRepeat
(XMLGEN_GET_ATTRVAL_LEN_HNDL_FCT fHndlFct)

Description

Function to register a handler function for the repeated delivering of an attribute value length. The handler function MUST return the same value as in the former call of the function registered in the function stxXMLGen_RegHndlFct_GetAttrValLen().

It can be necessary to freeze a value length, e.g. for calculating the exact content length in a request.

It is advisable to call this function before using XML generator.

Parameter

fHndlFct: *handler function (XMLGEN_GET_ATTRVAL_LEN_HNDL_FCT is defined in "xmlgen.h")*

Return Value

-

3.3.5 stxXMLGen_RegHndlFct_GetAttrVal

void stxXMLGen_RegHndlFct_GetAttrVal
(XMLGEN_GET_ATTRVAL_HNDL_FCT fHndlFct)

Description

Function to register a handler function for delivering an attribute value.

It is advisable to call this function before using XML generator.

Parameter

fHndlFct: handler function (XMLGEN_GET_ATTRVAL_HNDL_FCT is defined in "xmlgen.h")

Return Value

-

3.3.6 stxXMLGen_RegHndlFct_GetContentLen

void stxXMLGen_RegHndlFct_GetContentLen
(XMLGEN_GET_CONTENT_LEN_HNDL_FCT fHndlFct)

Description

Function to register a handler function for delivering a content length.

It is advisable to call this function before using XML generator.

Parameter

fHndlFct: handler function (XMLGEN_GET_CONTENT_LEN_HNDL_FCT is defined in

"xmlgen.h")

Return Value

-

3.3.7 stxXMLGen_RegHndlFct_GetContentLenRepeat

void stxXMLGen_RegHndlFct_GetContentLenRepeat (XMLGEN_GET_CONTENT_LEN_HNDL_FCT fHndlFct)

Description

Function to register a handler function for the repeated delivering of a content length. The handler function MUST return the same value as in the former call of the function registered in the function stxXMLGen_RegHndlFct_GetContentLen().

It can be necessary to freeze a value length, e.g. for calculating the exact content length in a request.

It is advisable to call this function before using XML generator.

Parameter

fHndlFct: handler function (XMLGEN_GET_CONTENT_LEN_HNDL_FCT is defined in "xmlgen.h")

Return Value

-

3.3.8 stxXMLGen_RegHndlFct_GetContent

void stxXMLGen_RegHndlFct_GetContent (XMLGEN_GET_CONTENT_HNDL_FCT fHndlFct)

Description

Function to register a handler function for delivering a content.

It is advisable to call this function before using XML generator.

Parameter

fHndlFct: handler function (XMLGEN_GET_CONTENT_HNDL_FCT is defined in "xmlgen.h")

Return Value

-

3.3.9 stxXMLGen_RegHndlFct_GetArrayCnt

void stxXMLGen_RegHndlFct_GetArrayCnt (XMLGEN_GET_ARRAY_CNT_HNDL_FCT fHndlFct)

Description

Function to register a handler function for delivering the count of array entries.

It is advisable to call this function before using XML generator.

Parameter

fHndlFct: handler function (XMLGEN_GET_ARRAY_CNT_HNDL_FCT is defined in "xmlgen.h")

Return Value

-

3.3.10 stxXMLGen_RegHndlFct_GetArrayCntRepeat

void stxXMLGen_RegHndlFct_GetArrayCntRepeat (XMLGEN_GET_ARRAY_CNT_HNDL_FCT fHndlFct)

Description

Function to register a handler function for the repeated delivering of the count of array entries. The handler function MUST return the same value as in the former call of the function registered in the function stxXMLGen_RegHndlFct_GetArrayCnt().

It can be necessary to freeze a value length, e.g. for calculating the exact content length in a request.

It is advisable to call this function before using XML generator.

Parameter

fHndlFct: handler function (XMLGEN_GET_ARRAY_CNT_HNDL_FCT is defined in "xmlgen.h")

Return Value

-

3.3.11 stxXMLGen_GetHeaderDataLen

UINT32_stx stxXMLGen_GetHeaderDataLen (void)

Description

Calculates the length of the XML header.

Parameter

-

Return Value

Length of the XML header

3.3.12 stxXMLGen_GetHeaderData

XML_ERR stxXMLGen_GetHeaderData (UINT8_stx uMode, STRING_stx szBuffer, UINT16_stx uLength)

Description

Gets the first/next packet of the XML header.

Parameter

uMode:	FIRST - first packet NEXT - next packet
szBuffer:	destination buffer (buffer where the packet is to be copied)
uLength:	buffer size

Return Value

One of the following values:

XML_ERR_INVALID_PARAM	parameter error
XML_ERR_NOT_READY	header is not yet copied completely (function must be called again)
XML_ERR_OK	header is copied completely (buffer ends with terminating zero)

3.3.13 stxXMLGen_GetCodeLen

UINT32_stx stxXMLGen_GetCodeLen
(XML_ENTRY FPTR_stx aXMLEntries)

Description

Calculates the length of XML code (without XML header) from a template.

Parameter

aXMLEntries:	XML template (s.a.)

Return Value

Length of XML code

3.3.14 stxXMLGen_GetCode

XML_ERR stxXMLGen_GetCode
(UINT8_stx uMode, XML_ENTRY FPTR_stx aXMLEntries, STRING_stx szBuffer, UINT16_stx uLength)

Description

Fills XML code (without XML header) from a template into buffer

Parameter

uMode:	FIRST - first packet NEXT - next packet
aXMLEntries:	XML template (s.a.)
szBuffer:	destination buffer (buffer where the packet is to be copied)
uLength:	buffer size

Return Value

One of the following values:

XML_ERR_NO_NAMESPACES	error in namespace declaration
XML_ERR_INVALID_PARAM	parameter error
XML_ERR_NOT_READY	header is not yet copied completely (function must be called again)
XML_ERR_OK	header is copied completely

(buffer ends with terminating zero)

3.4 Parser: Public functions

3.4.1 stxXMLPrs_Init

void stxXMLPrs_Init(void)

Description

Initializes the module.

The function must be called once before using any other XML parser functions.

Parameter

-

Return Value

-

3.4.2 stxXMLPrs_RegHndlFct_PutTagOpen

void stxXMLPrs_RegHndlFct_PutTagOpen (XMLPRS_PUT_TAG_OPEN_HNDL_FCT fHndlFct)

Description

Function to register a handler function for delivering an opened tag.

It is advisable to call this function before using XML parser.

Parameter

fHndlFct: handler function (XMLPRS_PUT_TAG_OPEN_HNDL_FCT is defined in "xmlprs.h")

Return Value

-

3.4.3 stxXMLPrs_RegHndlFct_PutAttrVal

void stxXMLPrs_RegHndlFct_PutAttrVal (XMLPRS_PUT_ATTR_VAL_HNDL_FCT fHndlFct)

Description

Function to register a handler function for delivering an attribute value.

It is advisable to call this function before using XML parser.

Parameter

fHndlFct: handler function (XMLPRS_PUT_ATTR_VAL_HNDL_FCT is defined in "xmlprs.h")

Return Value

-

3.4.4 stxXMLPrs_RegHndlFct_PutTagClose

void stxXMLPrs_RegHndlFct_PutTagClose
(XMLPRS_PUT_TAG_CLOSE_HNDL_FCT fHndlFct)

Description

Function to register a handler function for delivering an closed tag.

It is advisable to call this function before using XML parser.

Parameter

fHndlFct: handler function (XMLPRS_PUT_TAG_CLOSE_HNDL_FCT is defined in "xmlprs.h")

Return Value

-

3.4.5 stxXMLPrs_RegHndlFct_PutContentPkt

void stxXMLPrs_RegHndlFct_PutContentPkt
(XMLPRS_PUT_CONTENT_PKT_HNDL_FCT fHndlFct)

Description

Function to register a handler function for delivering a tag content.

It is advisable to call this function before using XML generator.

Parameter

fHndlFct: handler function (XMLPRS_PUT_CONTENT_PKT_HNDL_FCT is defined in "xmlprs.h")

Return Value

-

3.4.6 stxXMLPrs_RegHndlFct_ErrorHandler

void stxXMLPrs_RegHndlFct_ErrorHandler
(XMLPRS_ERROR_HNDL_FCT fHndlFct)

Description

Function to register a handler function for signalling an error.

Parameter

fHndlFct: handler function (XMLPRS_ERROR_HNDL_FCT is defined in "xmlprs.h")

Return Value

-

3.4.7 stxXMLPrs_ParseInit

void stxXMLPrs_ParseInit(void)

Description

Initialization routine for a parse process - must be called on every new parse process.

Parameter

-

Return Value

-

3.4.8 stxXMLPrs_Parse

XML_ERR stxXMLPrs_Parse
(STRING_stx szDataPacket, UINT16_stx uLength)

Description

Parses an XML packet.

Parameter

szDataPacket: pointer to XML data packet
uLength: length of XML data packet

Return Value

One of the following values:
XML_ERR_OK - ok
XML_ERR_PARSE - error

4 HTTP module

4.1 Overview

The HTTP module can be used to generate and parse HTTP header.

HTTP code can be generated in segments of arbitrary length.

The parse process works “on the fly”. That means that the HTTP header can be parsed immediately during the receive process.

The HTTP module is composed of three parts: a definition part, a generator and a parser.

The following files are included:

- httpdefs.c (main part: structures)
- httpdefs.h (main part: definitions)
- httpgen.c (generator: functions)
- httpgen.h (generator: function declarations, definitions)
- httpprs.c (parser: functions)
- httpprs.h (parser: function declarations, definitions)

The following public functions can be used from any HTTP controlling module. In the sevenstax library XML is controlled for instance by SOAP.

4.2 Generator: Public functions

4.2.1 stxHTTPGen_Init

void stxHTTPGen_Init(void)

Description

Initializes the module.

The function must be called once before using any other HTTP generator functions.

Parameter

-

Return Value

-

4.2.2 stxHTTPGen_SetMethod

void stxHTTPGen_SetMethod (UINT8_stx uMethodPara)

Description

Sets the used HTTP method.

Parameter

uMethodPara: Method to set, one of the following values:
HTTP_METHOD_OPTIONS
HTTP_METHOD_GET
HTTP_METHOD_HEAD
HTTP_METHOD_POST
HTTP_METHOD_PUT
HTTP_METHOD_DELETE
HTTP_METHOD_TRACE
HTTP_METHOD_CONNECT

Return Value

-

4.2.3 stxHTTPGen_SetResource

void stxHTTPGen_SetResource (STRING_stx szResourcePara)

Description

Sets the used HTTP resource.

Parameter

szResourcePara: Resource to set

Return Value

-

4.2.4 stxHTTPGen_SetContentLength

void stxHTTPGen_SetContentLength (UINT32_stx uContentLength)

Description

Sets the HTTP content length.

Parameter

uContentLength: Content length

Return Value

-

4.2.5 stxHTTPGen_SetContentType

void stxHTTPGen_SetContentType
(UINT8_stx uContentType, UINT8_stx uEncoding)

Description

Sets the HTTP content type and content encoding.

Parameter

uContentType: Content type, one of the following values:
HTTP_TYPE_TEXTXML

HTTP_TYPE_TEXTPLAIN

uEncoding: Content encoding, one of the following values:
HTTP_CHARSET_UTF8
HTTP_CHARSET_ISO8859_1

Return Value

-

4.2.6 stxHTTPGen_SetHost

void stxHTTPGen_SetHost
(STRING_stx szHost)

Description

Sets the HTTP host.

Parameter

szHost: Host name

Return Value

-

4.2.7 stxHTTPGen_SetUsernameAndPassword

void stxHTTPGen_SetUsernameAndPassword
(STRING_stx szUsername, STRING_stx szPassword)

Description

Sets username and password for HTTP authorization.

Parameter

szUsername: username

szPassword: *password*

Return Value

-

4.2.8 stxHTTPGen_GetSendHeaderData

void stxHTTPGen_GetSendHeaderData
(UINT8_stx uMode, STRING_stx szBuffer, UINT16_stx uLength)

Description

Function to register a handler function for delivering a content.

It is advisable to call this function before using XML generator.

Parameter

uMode: FIRST - first packet
NEXT - next packet
szBuffer: destination buffer (buffer where the packet is to be copied)
uLength: buffer size

Return Value

-

4.3 Parser: Public functions

4.3.1 stxHTTPPrs_Init

void stxHTTPPrs_Init(void)

Description

Initializes the module.

The function must be called once before using any other HTTP parser functions.

Parameter

-

Return Value

-

4.3.2 stxHTTPPrs_SetHndlFct_SetHeaderInfo

void stxHTTPPrs_SetHndlFct_SetHeaderInfo
(HTTP_SET_HEADER_INFO_FCT_TYPE pfnFct)

Description

Function to register a handler function for delivering the HTTP header fields with its values to the application.

It is advisable to call this function before using HTTP parser.

Parameter

pfnFct: handler function
(HTTP_SET_HEADER_INFO_FCT_TYPE is defined in "httpprs.h")

Return Value

-

4.3.3 stxHTTPPrs_ParseInit

void stxHTTPPrs_ParseInit(void)

Description

Initialization routine for a parse process - must be called on every new parse process.

Parameter

-

Return Value

-

4.3.4 stxHTTPPrs_Parse

HTTP_ERR stxHTTPPrs_Parse
(STRING_stx szDataPacket, UINT16_stx *puLength)

Description

Parses a HTTP data packet.

Parameter

szDataPacket: pointer to HTTP data packet
puLength: pointer to length of HTTP data packet
(will be set to position in packet where HTTP code has finished (if so))

Return Value

One of the following values:
HTTP_ERR_OK - ok
HTTP_ERR_PARSE - error

4.3.5 stxHTTPPrs_GetLastAnswerCode

UINT16_stx stxHTTPPrs_GetLastAnswerCode(void)

Description

Returns last received HTTP return code.

Parameter

-

Return Value

Last received HTTP return code

Copyright (c) 2009 by sevenstax GmbH
This document is an intellectual property of sevenstax GmbH. Unauthorized copying and distribution is prohibited.